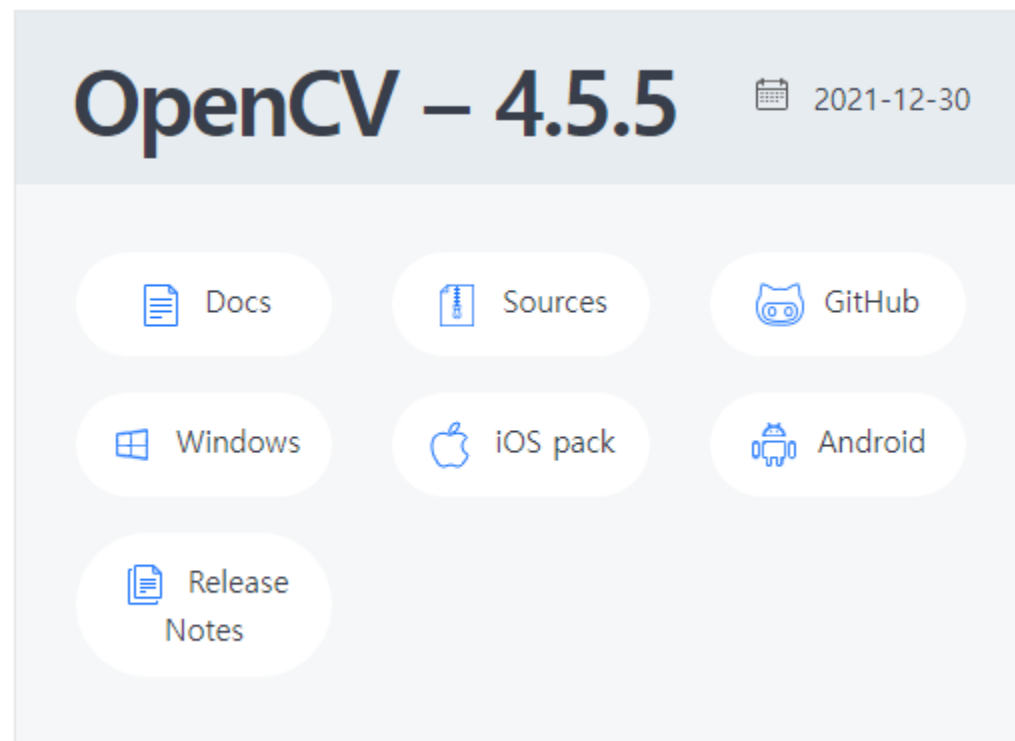


7. 영상 처리

이영주
young.kopo@gmail.com

Open CV 설치

```
>>> import cv2  
>>> print(cv2.__version__)  
4.5.5  
>>>
```



이미지 출력

```
import cv2

frame = cv2.imread('example.jpg', 1)
cv2.imshow('example Image', frame)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
>>> img = cv2.imread('lena.jpg', cv2.IMREAD_COLOR)
```

cv2.imread(fileName, flag)

이미지 파일을 flag값에 따라서 읽어들이입니다.

Parameters:

- **fileName (str)** - 이미지파일의 경로
- **flag (int)** - 이미지 파일을 읽을 때의 Option.

Returns: image객체 행렬

Return type: numpy.ndarray

이미지 읽기의 flag는 3가지가 있습니다.

- **cv2.IMREAD_COLOR** : 이미지 파일을 Color로 읽어들이입니다. 투명한 부분은 무시되며, Default값입니다.
- **cv2.IMREAD_GRAYSCALE** : 이미지를 Grayscale로 읽어 들입니다. 실제 이미지 처리시 중간단계로 많이 사용합니다.
- **cv2.IMREAD_UNCHANGED** : 이미지파일을 alpha channel까지 포함하여 읽어 들입니다.

ⓘ Note

3개의 flag대신에 1, 0, -1을 사용해도 됩니다.

이미지 출력

```
import cv2

frame = cv2.imread('example.jpg', 1)
cv2.imshow('example Image', frame)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

이미지 보기

`cv2.imshow()` 함수는 이미지를 사이즈에 맞게 보여줍니다.

```
>>> c22.imshow('image', img)
>>> cv2.waitKey(0)
>>> cv2.destroyAllWindows()
```

`cv2.imshow(title, image)`

읽어들인 이미지 파일을 윈도우창에 보여줍니다.

- Parameters:
- title (str) - 윈도우 창의 Title
 - image (numpy.ndarray) - `cv2.imread()` 의 return값

`cv2.waitKey()` 는 keyboard입력을 대기하는 함수로 0이면 key입력까지 무한대기이며 특정 시간동안 대기하려면 millisecond값을 넣어주면 됩니다.

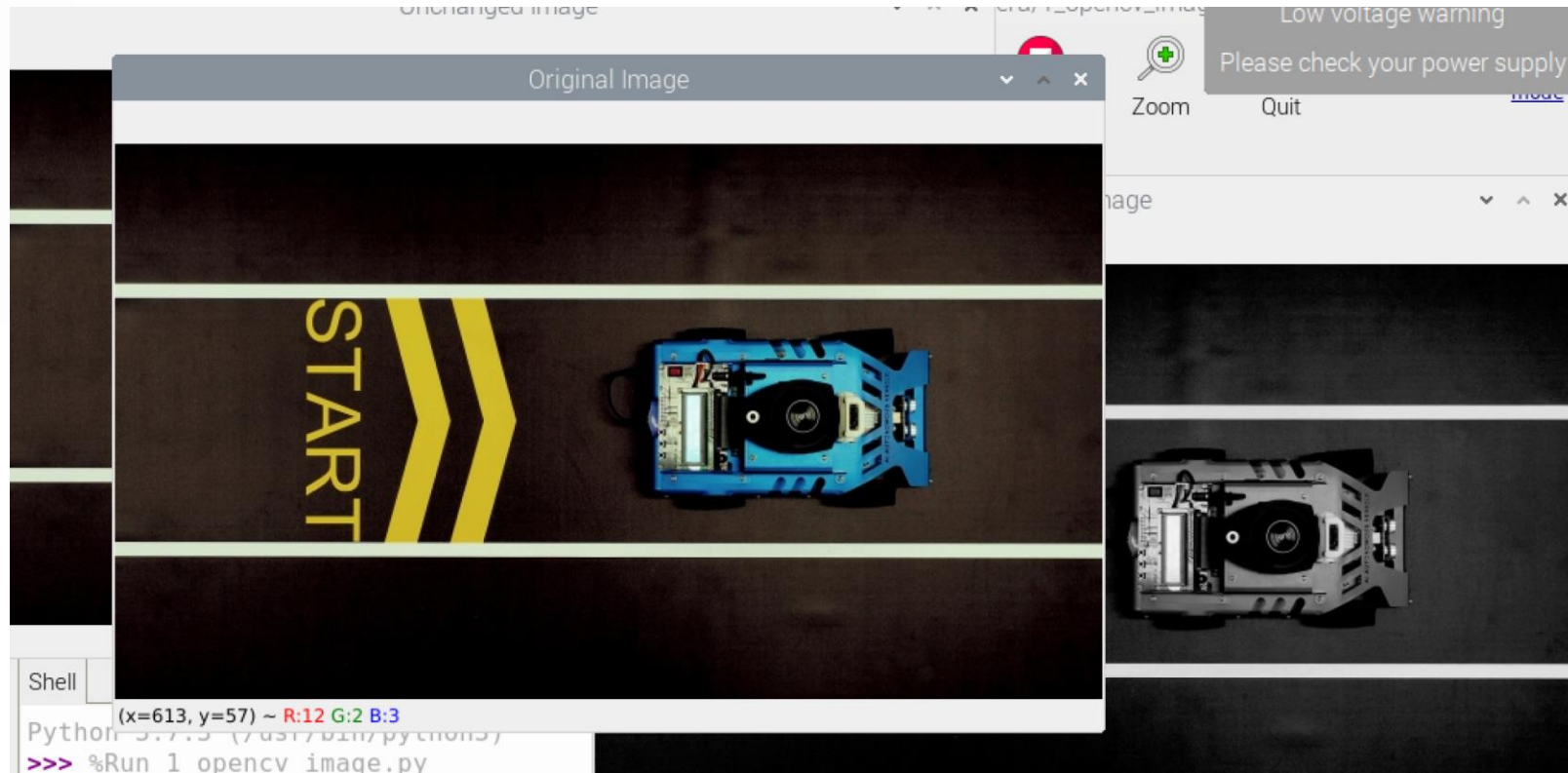
`cv2.destroyAllWindows()` 는 화면에 나타난 윈도우를 종료합니다. 일반적으로 위 3개는 같이 사용됩니다.

이미지 출력

```
import cv2

original = cv2.imread('example.jpg', 1)
gray = cv2.imread('example.jpg', 0)
unchange = cv2.imread('example.jpg', -1)

cv2.imshow('Original Image', original)
cv2.imshow('Gray Image', gray)
cv2.imshow('Unchanged Image', unchange)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

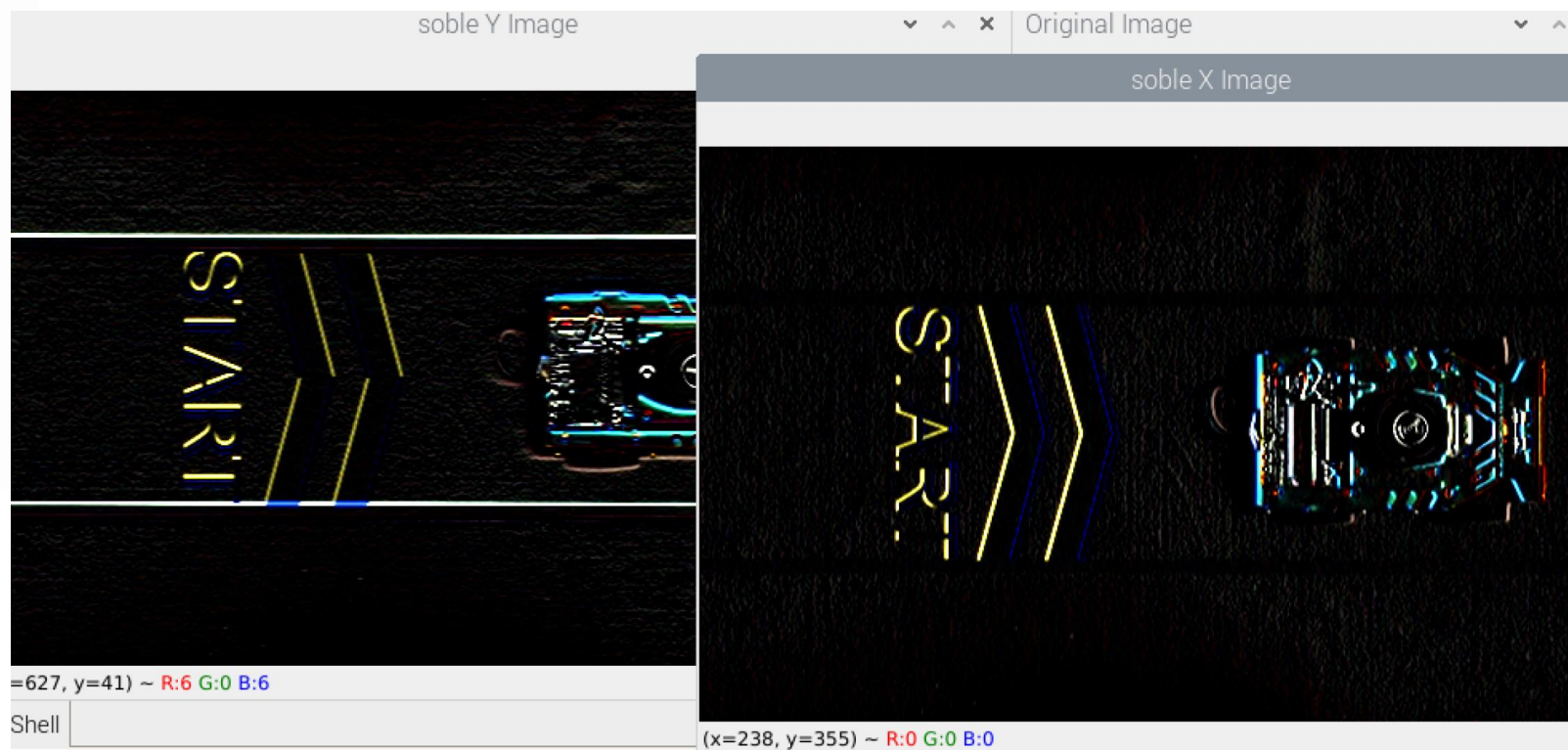


이미지 출력(필터, Sobel)

```
import cv2

original = cv2.imread('example.jpg', 1)
gray = cv2.imread('example.jpg', 0)
unchange = cv2.imread('example.jpg', -1)
dx = cv2.Sobel(original, -1, 1, 0)
dy = cv2.Sobel(original, -1, 0, 1)

cv2.imshow('Original Image', original)
cv2.imshow('soble X Image', dx)
cv2.imshow('soble Y Image', dy)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



◆ VideoCapture() [3/5]

```
cv::VideoCapture::VideoCapture ( const String & filename,  
                                int apiPreference  
                                )
```

Open video file or a capturing device or a IP video stream for video capturing with API Preference.

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

filename it can be:

- name of video file (eg. `video.avi`)
- or image sequence (eg. `img_%02d.jpg` , which will read samples like `img_00.jpg`, `img_01.jpg`, `img_02.jpg`, ...)
- or URL of video stream (eg. `protocol://host:port/script_name?script_params|auth`)
- or GStreamer pipeline string in gst-launch tool format in case if GStreamer is used as backend Note that each video stream or IP camera feed has its own URL scheme. Please refer to the documentation of source stream to know the right URL.

apiPreference preferred Capture API backends to use. Can be used to enforce a specific reader implementation if multiple are available: e.g.

`cv::CAP_FFMPEG` or `cv::CAP_IMAGES` or `cv::CAP_DSHOW`.

See also

`cv::VideoCaptureAPIs`

◆ isOpened()

virtual bool cv::VideoCapture::isOpened () const

virtual

Python:

cv.VideoCapture.isOpened() -> retval

Returns true if video capturing has been initialized already.

If the previous call to **VideoCapture** constructor or **VideoCapture::open()** succeeded, the method returns true.

Examples:

`samples/cpp/camshiftdemo.cpp`, `samples/cpp/facedetect.cpp`, `samples/cpp/laplace.cpp`, `samples/cpp/lkdemo.cpp`,
`samples/cpp/peopledetect.cpp`, `samples/cpp/polar_transforms.cpp`, `samples/cpp/segment_objects.cpp`, `samples/cpp/train_HOG.cpp`,
`samples/cpp/tutorial_code/videoio/video-write/video-write.cpp`, `samples/cpp/videowriter_basic.cpp`, and `samples/tapi/hog.cpp`.

◆ read()

virtual bool cv::VideoCapture::read (**OutputArray** image)

virtual

Python:

```
cv.VideoCapture.read( [ , image] ) -> retval, image
```

Grabs, decodes and returns the next video frame.

Parameters

[out] **image** the video frame is returned here. If no frames has been grabbed the image will be empty.

Returns

`false` if no frames has been grabbed

The method/function combines **VideoCapture::grab()** and **VideoCapture::retrieve()** in one call. This is the most convenient method for reading video files or capturing data from decode and returns the just grabbed frame. If no frames has been grabbed (camera has been disconnected, or there are no more frames in video file), the method returns false and the function returns empty image (with cv::Mat, test it with **Mat::empty()**).

Note

In **C API**, functions **cvRetrieveFrame()** and **cv.RetrieveFrame()** return image stored inside the video capturing structure. It is not allowed to modify or release the image! You can copy the frame using **cvCloneImage** and then do whatever you want with the copy.

Examples:

`samples/cpp/videowriter_basic.cpp`.

◆ release()

```
virtual void cv::VideoCapture::release ( )
```

virtual

Python:

```
cv.VideoCapture.release( ) -> None
```

Closes video file or capturing device.

The method is automatically called by subsequent **VideoCapture::open** and by **VideoCapture** destructor.

The C function also deallocates memory and clears *capture pointer.

카메라 영상 모니터링

```
import cv2
```

```
# 영상스트리밍 서비스인 MJPG-streamer를 쓰고 있기 때문에 URL로 접속한다.
```

```
cap = cv2.VideoCapture('http://127.0.0.1:8090/?action=stream')
```

```
# MJPG-streamer에서 비디오 디바이스인 /dev/video0를 쓰고 있기 때문에 아래 코드로는
```

```
# 동작하지 않으나 별도 서비스가 동작중이지 않을때는 cv2.VideoCapture(0) 를 쓴다.
```

```
# cap = cv2.VideoCapture(0)
```

```
while cap.isOpened:
```

```
    ret, frame = cap.read()
```

```
    cv2.imshow("example Video. (press 'q' to Quit)", frame)
```

```
    # 30프레임 만들기 위해 한 프레임 처리 후 33ms 대기한다.
```

```
    # 대기하는 동안 읽은 key는 key 변수에 저장한다.
```

```
    key = cv2.waitKey(33)
```

```
    # key 값이 q가 들어오면 종료한다.
```


```
    if key == ord('q'):
```


```
        break
```

```
cap.release()
```

```
cv2.destroyAllWindows()
```

새 프로젝트

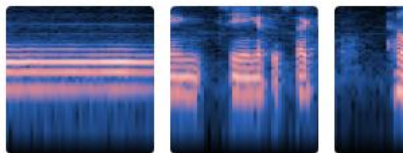
 Drive에 있는 기존 프로젝트를 엽니다.

 파일에서 기존 프로젝트를 엽니다.



이미지 프로젝트

파일 또는 웹캠에서 가져온 이미지를 기반으로 학습시키세요.



오디오 프로젝트

파일 또는 마이크에서 가져온 1초 분량의 사운드를 기반으로 학습시키세요.



포즈 프로젝트

파일 또는 웹캠에서 가져온 이미지를 기반으로 학습시키세요.

더 다양한 기능이 곧 제공될 예정입니다.

더 많은 모델이 개발되면 여기에 표시될 예정입니다.

Teachable Machine 활용하기

- 티쳐블 머신 웹 페이지에 접속한다
- 구글 검색 '티쳐블머신' 혹은 아래 주소로 접근한다.

<https://teachablemachine.withgoogle.com/>



- 상단과 웹 페이지 정면에 있는 파란색 '시작하기' 버튼을 누른다.

티쳐블 머신 활용

텐서플로우 예제 수행

1_keras/1_keras_run.py 실행

```
[[2.3148548e-09 3.9140223e-06 9.9999607e-01]]
```

```
Windows PowerShell
2022-07-22 06:52:18.148558: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libra
ry 'cudart64_110.dll'; dLError: cudart64_110.dll not found
2022-07-22 06:52:18.149166: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libra
ry 'cublas64_11.dll'; dLError: cublas64_11.dll not found
2022-07-22 06:52:18.149710: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libra
ry 'cublaslt64_11.dll'; dLError: cublaslt64_11.dll not found
2022-07-22 06:52:18.150361: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libra
ry 'cufft64_10.dll'; dLError: cufft64_10.dll not found
2022-07-22 06:52:18.150845: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libra
ry 'curand64_10.dll'; dLError: curand64_10.dll not found
2022-07-22 06:52:18.151322: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libra
ry 'cusolver64_11.dll'; dLError: cusolver64_11.dll not found
2022-07-22 06:52:18.151834: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libra
ry 'cuspars64_11.dll'; dLError: cuspars64_11.dll not found
2022-07-22 06:52:18.152311: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libra
ry 'cudnn64_8.dll'; dLError: cudnn64_8.dll not found
2022-07-22 06:52:18.152521: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1835] Cannot dlopen some GPU libraries. P
lease make sure the missing libraries mentioned above are installed properly if you would like to use GPU. Follow the gu
ide at https://www.tensorflow.org/install/gpu for how to download and setup the required libraries for your platform.
Skipping registering GPU devices...
2022-07-22 06:52:18.153976: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized wit
h oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations:
 AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
WARNING:tensorflow:No training configuration found in the save file, so the model was *not* compiled. Compile it manuall
y.
2022-07-22 06:52:19.848483: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR Optimizatio
n Passes are enabled (registered 2)
[[2.3148548e-09 3.9140223e-06 9.9999607e-01]]
PS C:\Users\zzub0\OneDrive\바탕 화면\TF_Keras\1_Keras>
```



Teachable Machine 활용하기

- 오늘 진행할 프로젝트는 '이미지 프로젝트' 이다.
- 이미지 프로젝트를 클릭하도록 한다.



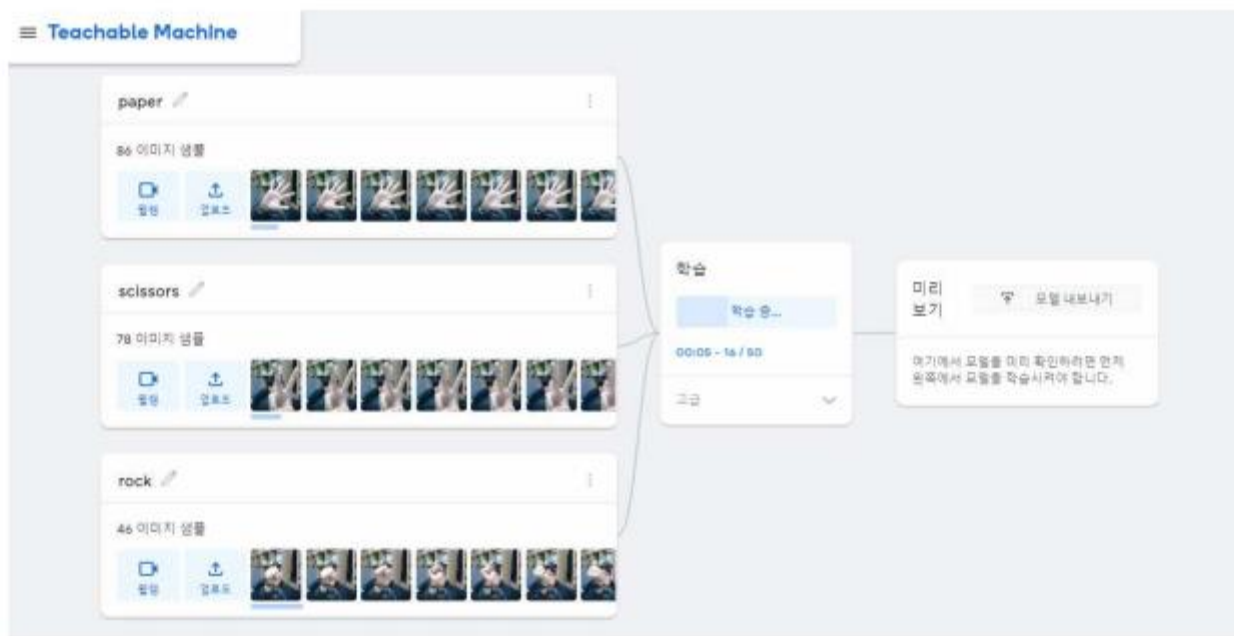
Teachable Machine 활용하기

- 두가지 선택 항목 중 **표준 이미지 모델**을 진행한다.
- **삽입된 이미지 모델** 은 특정 MCU에 사용하는 용도의 모델을 제작한다.



Teachable Machine 활용하기

- 가위-바위-보 기준으로 3개 클래스의 이미지를 수집했다.
- 수집이 완료되면 학습하기 버튼으로 학습을 수행한다.
- 세부 옵션 조정이 가능하지만 현재는 진행하지 않는다.




티쳐블 머신 활용

탭을 전환하지 마세요.
모델을 학습시키려면 이 탭이 열려 있어야 합니다.

[다시 표시하지 않음](#) [확인](#)

paper 


124 이미지 샘플


scissors 

107 이미지 샘플

rock 

118 이미지 샘플

학습

 학습 중...

00:03 - 10 / 50

고급 

미리 보기

 모델 내보내기

여기에서 모델을 미리 확인하려면 먼저 왼쪽에서 모델을 학습시켜야 합니다.

티쳐블 머신 활용

paper 124 이미지 샘플

scissors 107 이미지 샘플

rock 118 이미지 샘플

학습

모델 학습 완료됨

고급

클래스 추가

미리 보기

모델 내보내기

입력 사용 Webcam

출력

paper

scissors 44%

rock 64%

프로젝트에서 모델을 사용하려면 모델을 내보내세요.

Tensorflow.js Tensorflow Tensorflow Lite

모델 변환 유형:

☒ Keras ☐ Savedmodel

모델 다운로드

모델을 keras.h5 모델로 변환합니다. 변환은 클라우드에서 이루어지지만, 학습 데이터는 업로드되지 않으며 학습이 완료된 모델만 업로드됩니다.

모델에서 사용할 코드 스니펫:

Keras OpenCV Keras Github에 참여

```
from keras.models import load_model # TensorFlow is required for Keras to work
from PIL import Image, ImageOps # Install pillow instead of PIL
import numpy as np

# Disable scientific notation for clarity
np.set_printoptions(suppress=True)

# Load the model
model = load_model("keras_Model.h5", compile=False)

# Load the labels
class_names = open("labels.txt", "r").readlines()

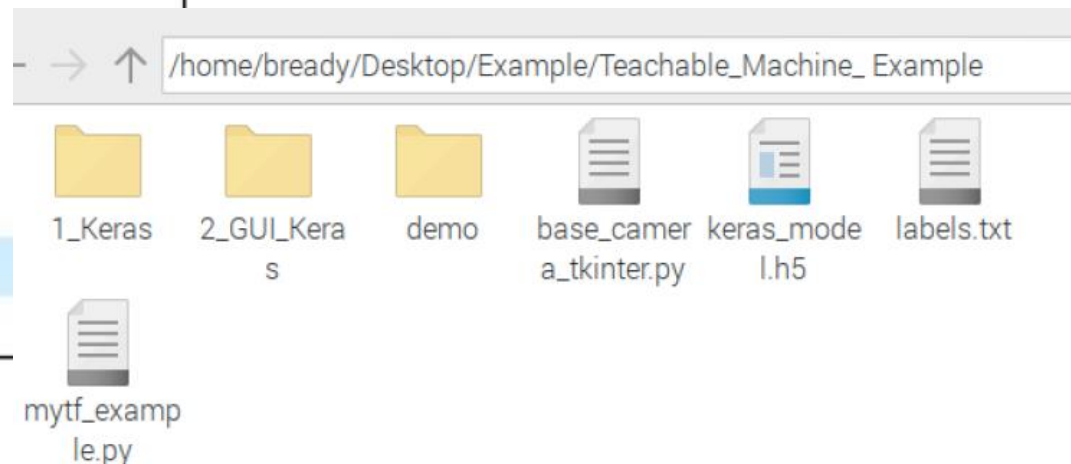
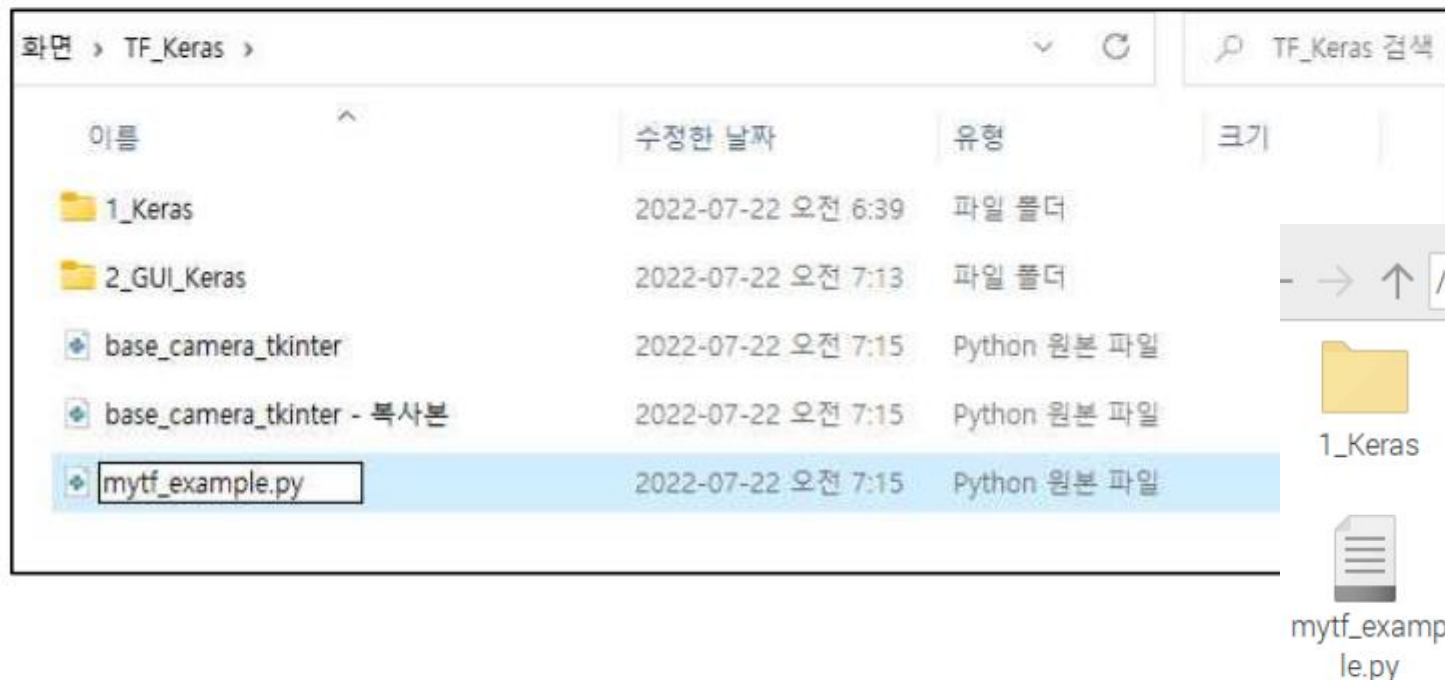
# Create the array of the right shape to feed into the keras model
# The 'length' or number of images you can put into the array is
# determined by the first position in the shape tuple, in this case 1
data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
```

클래스 추가

티쳐블 머신 활용(AI 인식)

AI 인식기 만들기

- tkinter를 이용하여 카메라를 출력하는 예제를 수정한다.
- 예제 디렉토리에 있는 TF_Keras/base_camera_tkinter.py 를 수정한다.
- base_camera_tkinter.py를 하나 복사하여 이름을 변경한다.
- 예제에서는 mytf_example.py로 수정



분류 인식기 만들기 : 인식 값 표시를 위한 label 추가

- 30~33 라인에 label 및 label 데이터 변수를 추가한다.
- label은 msg라는 이름으로 생성해서 pack을 사용해서 배치한다.

```
22  try:
23      ...root = Tk() .....
24      ...root.title('Camera')
25      ...root.geometry("640x520+10+10")
26
27      ...lmain = Label(root)
28      ...lmain.pack()
29
30      ...value = StringVar()
31      ...value.set("텍스트")
32      ...msg = Label(root, background="yellow", textvariable = value)
33      ...msg.pack()
34
35      ...cap = cv2.VideoCapture(0)
36      ...show_frame()
37      ...root.mainloop() .....
```

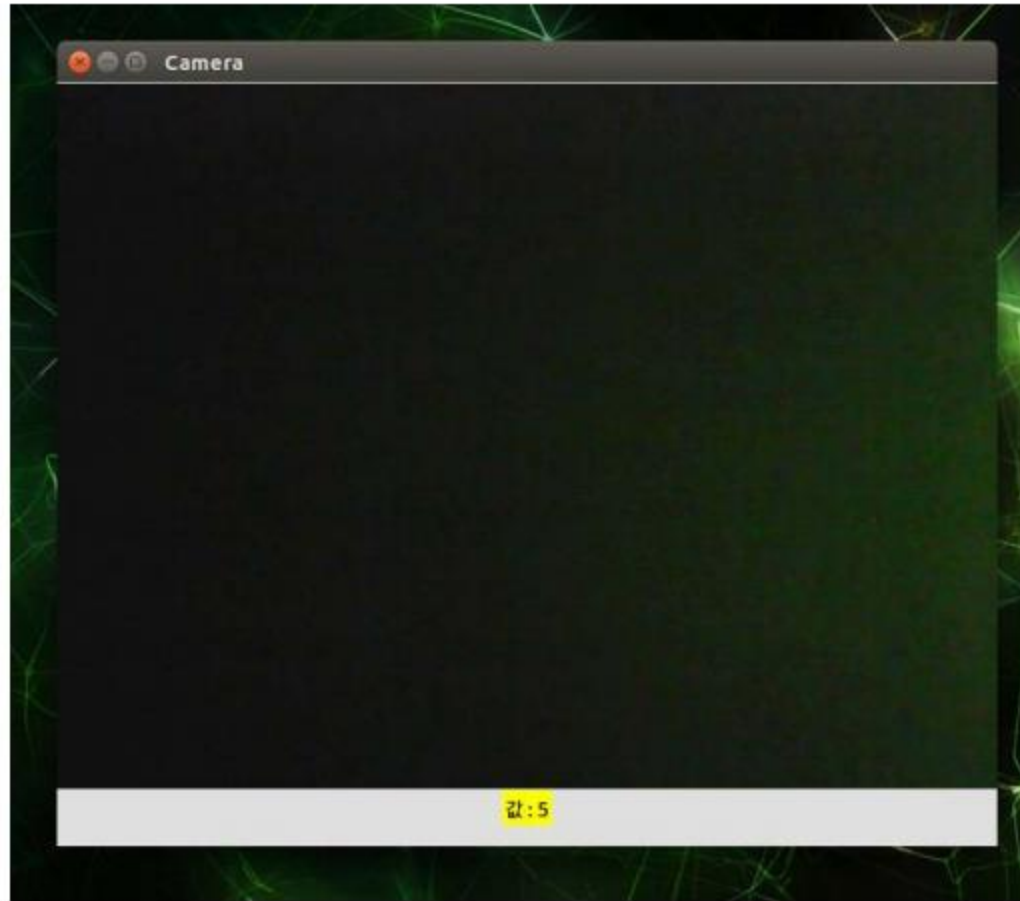

분류 인식기 만들기 : 인식 값 표시를 위한 label 추가

- 6번 라인은 랜덤 함수를 사용하기 위한 import 이다.
- 7번 라인은 해당 파이썬 프로그램에서 전역적으로 사용하는 값 저장 변수이다.
- 18~19번은 랜덤 함수로 임의의 값 (차후 인식 데이터 값) 을 저장 한 후, 레이블의 데이터 변수인 value의 값을 변경시켜 label에 변경된 값이 보여지도록한다.

```
6  import random
7
8  value= None
9
10 def show_frame():
11     ret, frame = cap.read()
12     processImage = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
13     processImage = cv2.resize(processImage, (640, 480))
14     processImage = Image.fromarray(processImage)
15     processImage = ImageTk.PhotoImage(image=processImage)
16     lmain.processImage = processImage
17     lmain.configure(image=processImage)
18     prediction = str(random.randrange(1,10))
19     value.set("값 : " + prediction )
20     lmain.after(1, show_frame)
```


티쳐블 머신 활용(AI 인식)

분류 인식기 만들기 : 인식 값 표시를 위한 label 추가



티쳐블 머신 활용(AI 인식)

분류 인식기 만들기 :

- 카메라의 영상을 받아 인식 결과 출력
- 단순히 영상 뿐만이 아니라 인식된 영상을 통해 인식 결과를 출력한다



티쳐블 머신 활용(AI 인식)

분류 인식기 만들기 : 카메라 영상을 받아 인식 결과 출력

- 1_keras_run.py에 있던 import 내용을 추가한다.
- 이전에 있던 random은 더 이상 쓰지 않기 때문에 없애도 상관 없다.

```
1  from tkinter import *
2  from PIL import Image, ImageTk, ImageFilter
3  import time
4  import threading
5  import cv2
6  from tensorflow.keras.models import load_model
7  import numpy as np
8  from PIL import Image, ImageOps
9
10 value= None
```

분류 인식기 만들기 : 카메라 영상을 받아 인식 결과 출력

- ex1_tf_keras.py에 있던 내용중 선언 및 정의에 해당하는 부분을 상단에 배치한다.

```
9
10 value= None
11
12 model = load_model('keras_model.h5')
13 data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
14 size = (224, 224)
15
16 def show_frame():
17     ret, frame = cap.read()
18     processImage = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

티쳐블 머신 활용(AI 인식)

분류 인식기 만들기 : 카메라 영상을 받아 인식 결과 출력

- 20번 라인에서 카메라의 이미지를 받아 티쳐블 머신이 사용하는 224x224 사이즈로 줄인다.
- 28 ~ 29라인에서 ex1_tf_keras 예제에서 이미지 경로로부터 이미지를 읽듯 카메라의 이미지 포맷을 사용할 수 있는 포맷으로 변환한다.

```
16 def show_frame():
17     ret, frame = cap.read()
18     processImage = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
19
20     recog = cv2.resize(processImage, (224, 224))
21
22     processImage = cv2.resize(processImage, (640, 480))
23     processImage = Image.fromarray(processImage)
24     processImage = ImageTk.PhotoImage(image=processImage)
25     lmain.processImage = processImage
26     lmain.configure(image=processImage)
27
28     recog = Image.fromarray(recog)
29     recog = ImageOps.fit(recog, size, Image.ANTIALIAS)
30     image_array = np.asarray(recog)
31     normalized_image_array = (image_array.astype(np.float32) / 127.0) - 1
32     data[0] = normalized_image_array
33     prediction = model.predict(data)
34     value.set(str(prediction))
35     lmain.after(1, show_frame)
```

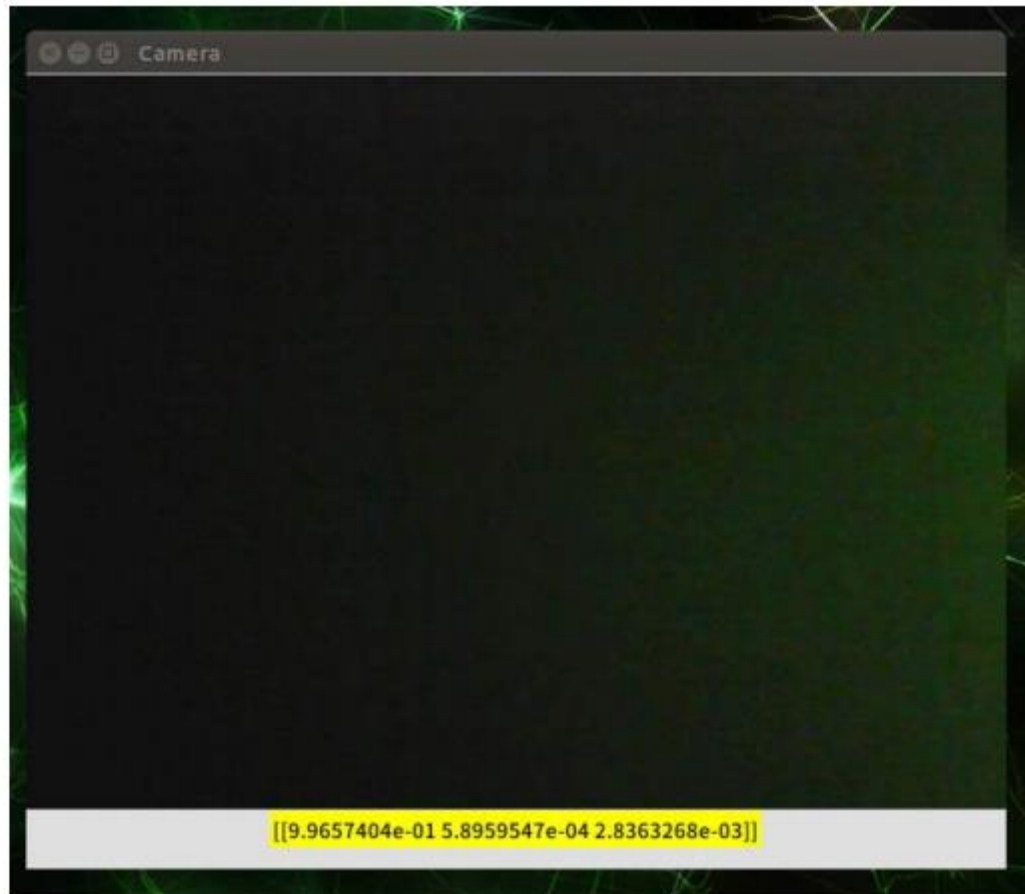

분류 인식기 만들기 : 카메라 영상을 받아 인식 결과 출력

- ex1_tf_keras 예제와 마찬가지로 읽어온 이미지를 토대로 추론한 결과를 도출하고
- 해당 결과를 앞서 생성했던 label에 표시하여 최종적으로 결과를 확인할 수 있다.

```
16 def show_frame():
17     ret, frame = cap.read()
18     processImage = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
19
20     recog = cv2.resize(processImage, (224, 224))
21
22     processImage = cv2.resize(processImage, (640, 480))
23     processImage = Image.fromarray(processImage)
24     processImage = ImageTk.PhotoImage(image=processImage)
25     lmain.processImage = processImage
26     lmain.configure(image=processImage)
27
28     recog = Image.fromarray(recog)
29     recog = ImageOps.fit(recog, size, Image.ANTIALIAS)
30     image_array = np.asarray(recog)
31     normalized_image_array = (image_array.astype(np.float32) / 127.0) - 1
32     data[0] = normalized_image_array
33     prediction = model.predict(data)
34     value.set(str(prediction))
35     lmain.after(1, show_frame)
```

티쳐블 머신 활용(AI 인식)

분류 인식기 만들기 : 카메라 영상을 받아 인식 결과 출력



티쳐블 머신 활용(AI 인식 + 인식률(추론데이터 출력))

레이블 값 읽기 함수 추가

```
def readLabels():  
    try:  
        f = open("labels.txt", 'r')  
        list_labels = []  
        while True:  
            line = f.readline()  
            if not line: break  
            getlabel = line.split(' ')  
            getlabel = getlabel[1].split('\n')  
            list_labels.append(getlabel[0])  
        f.close()  
    except Exception as e :  
        print(e)  
    return list_labels
```

```
def show_frame():  
    imglabel = readLabels()  
  
    ret, frame = cap.read()
```

티쳐블 머신 활용(AI 인식 + 인식률(추론데이터 출력))

라벨값 추가

```
normalized_image_array = (image_array.astype(np.float32) / 127.0) - 1
data[0] = normalized_image_array
prediction = model.predict(data)

obj = []
for i in prediction[0]:
    v = int(float(i)*1000)/10
    obj.append(v)
value.set(imglabel[int(obj.index(max(obj)))] + " " + str(max(obj)) + " %")

#prediction = str(random.randrange(1,10))
#value.set(str(prediction))
lmain.after(1, show_frame)

try:
    root = Tk()
    root.title('Camera')
    root.geometry("640x520+10+10")
```

마스크 착용 여부 감지

```
import tensorflow as tf
from scripts import detection_color
import numpy as np
import cv2
import os
import time
import threading

MODEL_PATH = "training_models"
```

인공지능 동작 수행을 위해 'tensorflow' 모듈을 'tf'로 추가한다.

감지 결과의 색상을 다르게 적용하기 위해 'detection_color' 모듈을 추가한다.

스레드 함수를 사용하기 위해 추가

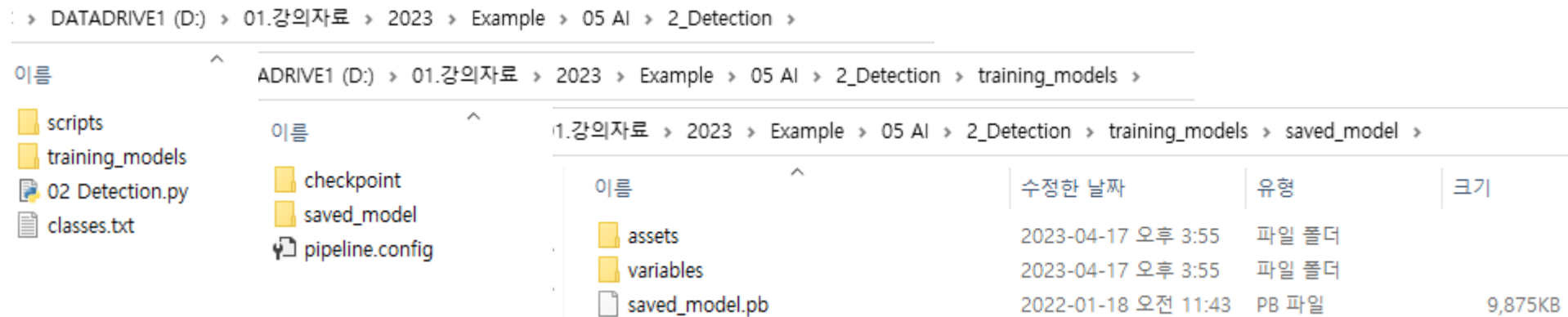
객체감지 인공지능 모델이 저장된 경로를 설정 인공지능 모델이 'training_models' 폴더에 저장

```
CLASS_FILE_PATH = "classes.txt"
MODEL_FILE_PATH = os.path.join(MODEL_PATH, "saved_model")
```

classes.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

without_mask
with_mask



마스크 착용 여부 감지

```
def _reader(camera):                                # 카메라에서 영상 이미지를 읽어 오는 함수 정의
    global ret, frame                                # 전역변수 선언
    while(True):
        ret, frame = camera.read()
        if(ret == False or frame is None):
            break
        time.sleep(0.001)

camera = cv2.VideoCapture('http://127.0.0.1:8090/?action=stream')
read_thread = threading.Thread(target=_reader, args=(camera,))
read_thread.daemon = True
read_thread.start()                                #쓰레드 실행( _reader() 함수 실행)
```

마스크 착용 여부 감지

```
model = tf.compat.v2.saved_model.load(MODEL_FILE_PATH, tags=[tf.compat.v2.saved_model.SERVING]) # 모델 불러와 model 객체에 저장
category_index = open(CLASS_FILE_PATH, 'r').read().splitlines() #클래스 파일에서 클래스 이름 가져옴. Category_index에 저장
color_list = detection_color.ColorList
color_list = [color_list[(i - 1) % len(color_list)]['code'] for i in range(len(color_list))]
print("추론 진행 중 (클래스가 처음 만들어질 때만 1번 실행)")
tf_input_tensor_name : np.zeros((1, 300, 300, 3))
def execute(frame):
    frame_det = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    frame_det = cv2.resize(frame_det, (300, 300), interpolation=cv2.INTER_AREA)
    try:
        output_dict = model(frame_det[None, ...])
        scores = output_dict["detection_scores"][0].numpy()
        boxes = output_dict["detection_boxes"][0].numpy()
        classes = output_dict["detection_classes"][0].numpy().astype(int)
    except (tf.errors.UnknownError, tf.errors.ResourceExhaustedError, tf.errors.InternalError) as e:
        print(e)
        print("GPU 메모리 부족 혹은 다른 원인으로 인해 작업이 중단되었습니다.")
        return
    class_names = np.array([category_index[cls_num-1] for cls_num in classes])
    draw_colors = np.array([color_list[cls_num-1] for cls_num in classes])
```

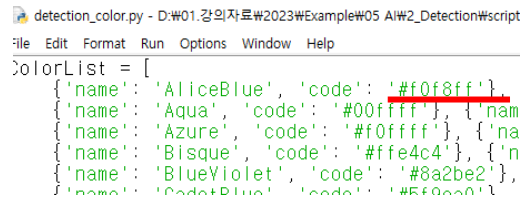
#검출객체의 확률, 영역, 클래스스배열변수에 저장

```
percentage = 0.5
mask = np.where(scores >= percentage)[0]
scores = scores[mask]
boxes = boxes[mask]
classes = classes[mask]
class_names = class_names[mask]
draw_colors = draw_colors[mask]
num_detections = len(mask)
return (scores, boxes, classes, class_names, draw_colors, num_detections)
```

마스크 착용 여부 감지

try:

```
while(True):
    original_frame = frame.copy() #frame 복사
    if(not ret): # 카메라 연결되지 않을경우 종료
        break
    scores, boxes, classes, class_names, draw_colors, num_detections = execute(frame)
    # 객체감지 추론을 통해 확률, 영역, 클래스, 클래스 이름, 감지색상, 감지 수량 리턴
    box = boxes * np.array([frame.shape[0], frame.shape[1], frame.shape[0], frame.shape[1]])
    box = np.round(box).astype(int)
    fontScale = frame.shape[0] * 1.3e-3
    topFontScale = frame.shape[0] * 1.5e-3
    for i in range(num_detections): #감지된 수량만큼 반복
        (y1, x1, y2, x2) = box[i]
        boxW = x2 - x1
        boxH = y2 - y1
        rgb_color = hexToRGB(draw_colors[i])
        bgr_color = rgb_color[::-1]
        cv2.rectangle(
            img = frame,
            pt1 = (x1, y1),
            pt2 = (x2, y2),
            color = bgr_color,
            thickness = 4,
        )
        text_label = "{0}:{1:.1%}".format(class_names[i], scores[i])
        x, y = (int(x1 + 3), int(y1 + 13)) if True else (int(x1), int(y1))
        size = cv2.getTextSize(text_label, cv2.FONT_HERSHEY_SIMPLEX, fontScale, 4)[0]
```



```
detection_color.py - D:\#01 강의자료\#2023\#Example\#05 AI\#2_Detection\#script
File Edit Format Run Options Window Help
colorList = [
    {'name': 'AliceBlue', 'code': '#f0f8ff'},
    {'name': 'Aqua', 'code': '#00ffff'},
    {'name': 'Azure', 'code': '#f0ffff'},
    {'name': 'Bisque', 'code': '#ffe4c4'},
    {'name': 'BlueViolet', 'code': '#8a2be2'},
    {'name': 'CadetBlue', 'code': '#5f9e9d'}
```

```
cv2.rectangle(
    img = frame,
    pt1 = (x, y - size[1]),
    pt2 = (x + size[0], y),
    color = bgr_color,
    thickness = cv2.FILLED
)
cv2.putText(
    img = frame,
    text = text_label,
    org = (x, y),
    fontFace = cv2.FONT_HERSHEY_SIMPLEX,
    fontScale = fontScale,
    color = (0, 0, 0),
    thickness = 2
)
frame = cv2.addWeighted(original_frame, 0.4, frame, 1 - 0.4, 0)
cv2.imshow('Object Detection', frame)
waitKey = cv2.waitKey(1)
if waitKey == ord('q'):
    break

except KeyboardInterrupt:
    pass
except Exception as e:
    import traceback
    traceback.print_exc()
    print(e)
finally:
    cv2.destroyAllWindows()
    print("종료")
```

이미지 분류 여부 감지(사과/바나나)

```
import tensorflow as tf
import tensorflow.keras.applications as applications
```

```
import numpy as np
import cv2
import os
import time
import threading
```

```
# 학습 모델 저장 경로
MODEL_PATH = "training_models"
```

```
MODEL_H5_PATH = os.path.join(MODEL_PATH, "classification_model.h5")
LABELS_PATH = os.path.join(MODEL_PATH, "classes.txt")
```

```
# 카메라 설정
def _reader(camera):
    global ret, frame

    while(True):
        ret, frame = camera.read()
```

```
        if(ret == False or frame is None):
            break
        time.sleep(0.001)
```

```
camera = cv2.VideoCapture('http://127.0.0.1:8090/?action=stream')
```

```
read_thread = threading.Thread(target=_reader, args=(camera,))
read_thread.daemon = True
read_thread.start()
```

```
# 모델 불러오기
model = tf.keras.models.load_model(MODEL_H5_PATH)
```

```
# 클래스 이름 가져오기
category_index = open(LABELS_PATH, 'r').read().splitlines()
```

```
# 이미지 정규화 방법을 가져옵니다.
if("mobilenet" in model.name):
    preprocessor = applications.mobilenet.preprocess_input
else:
    preprocessor = lambda x: x
```

```
print("이미지 추론 진행 중 (클래스가 처음 만들어질 때만 1번 실행)")
```

```
# 처음 추론 시 생기는 딜레이 방지
tf_input_tensor_name: np.zeros((1, 224, 224, 3))
```

```
# 이미지 분류를 진행하는 함수
def execute(frame):
    frame_cls = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    frame_cls = cv2.resize(frame_cls, (224, 224), interpolation=cv2.INTER_AREA)
    frame_cls = preprocessor(frame_cls)
```

```
    scores = model.predict(frame_cls[None, ...])
    scores = np.squeeze(scores)
```

```
    top_k = scores.argsort()[-5:][::-1]
```

```
    result_label = category_index[top_k[0]]
    result_accuracy = scores[top_k[0]]
```

```
    return (result_label, result_accuracy)
```


이미지 분류 여부 감지(사과/바나나)

```
try:
    while(True):
        if(not ret):
            break

        classes_name, scores = execute(frame)

        text_label = classes_name + " : " + str(round(scores * 100, 2)) + "%"

        cv2.putText(
            img = frame,
            text = text_label,
            org = (30, 40),
            fontFace = cv2.FONT_HERSHEY_SIMPLEX,
            fontScale = frame.shape[0] * 1.5e-3,
            color = (255, 0, 255),
            thickness = 2,
            lineType = cv2.LINE_AA
        )
        cv2.imshow('Classification', frame)
        waitKey = cv2.waitKey(1)
        if waitKey == ord('q'):
            break
except KeyboardInterrupt:
    pass
except Exception as e:
    import traceback
    traceback.print_exc()
    print(e)
finally:
    cv2.destroyAllWindows()
    print("종료")
camera.release()
tf.keras.backend.clear_session()
```